



暨南大學  
JINAN UNIVERSITY

# Course Selecting System

(First semester for Year 2024-2025)

Course Name: Database System

Course Type: Optional

Student Name: 段瑞豪

Student ID: 2022180163

College: International School

Department: \_\_\_\_\_

Major: Computer Science and Technology

Professor: 吴汉瑞

**December 11<sup>th</sup>, 2024**

## Contents

1. Introduction.....	3
2. System Features and Objectives .....	3
3. Entity-Relationship (ER) Model Overview.....	4
4. System Usage Guidelines for mySQL workbench .....	5
4.1 Viewing Available Courses.....	5
4.2 Registering for Courses .....	6
4.3 Dropping a Course.....	6
4.4 Viewing Registered Courses .....	6
5. System Usage Guidelines for the Python code.....	7
6. Testing and Validation .....	11
7. Conclusion .....	15
8. Future Enhancements .....	15
9. References.....	16

# 1. Introduction

The Course Registration System is an efficient and interactive platform developed to assist students in enrolling in courses while ensuring that academic constraints such as prerequisite verification, credit limits, and schedule conflicts are respected. This report outlines the system's usage guidelines, detailing its features, structure, and operational processes.

The system is built using a relational database model to manage essential data, including student records, course information, and course selections. It prioritizes user-friendly interaction, data integrity, and reliability.

## 2. System Features and Objectives

The Course Registration System aims to achieve the following objectives:

- Enable students to view, select, and drop courses seamlessly.
- Ensure prerequisites are completed before advanced course enrollment.
- Prevent students from exceeding a 21-credit limit per semester.
- Detect time conflicts among selected courses.

The system utilizes MySQL for database management and Python for backend logic, ensuring smooth functionality and precise validation processes.

### 3. Entity-Relationship (ER) Model Overview

The system employs an Entity-Relationship model consisting of three main entities:

- **Students:** Represents student records, including their unique StudentID and name.
- **Courses:** Contains detailed course information such as CourseID, CourseName, credits, prerequisites, and time slots.
- **Selections:** Acts as a bridge table connecting students to their chosen courses. It includes attributes like StudentID and CourseID to manage course registration.

This relational model supports constraints such as prerequisite verification, credit limits, and conflict detection, ensuring accurate and consistent data management.

+-----+	+-----+	+-----+
Students	Selections	Courses
+-----+	+-----+	+-----+
StudentID (PK)  <----->  SelectionID (PK)  <----->  CourseID (PK)		
Name	StudentID (FK)	CourseName
Email	CourseID (FK)	Credits
EnrollmentDate	EnrollmentDate	Prerequisite (FK)
+-----+	+-----+	TimeSlot
		+-----+

Figure: ER model for the course selection system

The Entity Relationship (ER) model for the course registration system consists of three main entities: *\*Students\**, *\*Courses\**, and *\*Selections\**. The *\*Students\** entity represents the users of the system and includes attributes such as 'StudentID' (the primary key), 'Name', 'Email', and 'EnrollmentDate'. The *\*Courses\** entity stores all course-related information, with attributes including 'CourseID' (the primary key), 'CourseName', 'Credits', 'TimeSlot', and a self-referencing foreign key, 'Prerequisite', which allows a course to define another course as its prerequisite. The relationship between students and courses is established through the *\*Selections\** entity, which serves as a bridge table to implement the many-to-many relationship. The *\*Selections\** table includes a primary key 'SelectionID', as well as foreign keys 'StudentID' (referencing the *\*Students\** table) and 'CourseID' (referencing the *\*Courses\** table).

The model is designed to ensure logical relationships and constraints within the system. For instance, the recursive relationship in the *\*Courses\** table allows courses to depend on other courses as prerequisites, enabling checks for eligibility before enrollment. Similarly, the *\*Selections\** table ensures that a student can enroll in multiple courses while allowing multiple students to register for the same course. Logical constraints such as credit limits and time conflicts are implemented at the application level, ensuring that the total credits selected by a student do not exceed a specified limit (e.g., 21) and that there are no overlaps in course time

slots. Together, these entities and relationships form a robust model that supports course management, prerequisite checks, and student registrations effectively.

## Constraints

To ensure system logic is enforced, add the following constraints:

- Foreign Keys:**
  - Selections.StudentID references Students.StudentID.
  - Selections.CourseID references Courses.CourseID.
  - Courses.Prerequisite references Courses.CourseID.
- Credit Limit Constraint** (Enforced in Application Logic):
  - Sum of credits for selected courses must not exceed 21.
- Time Conflict Constraint** (Enforced in Application Logic):
  - A student cannot enroll in courses with overlapping TimeSlot values.
- Check Prerequisite:**
  - Before inserting into Selections, verify the student has completed the prerequisite course.

## 4. System Usage Guidelines for mySQL workbench

### 4.1 Viewing Available Courses

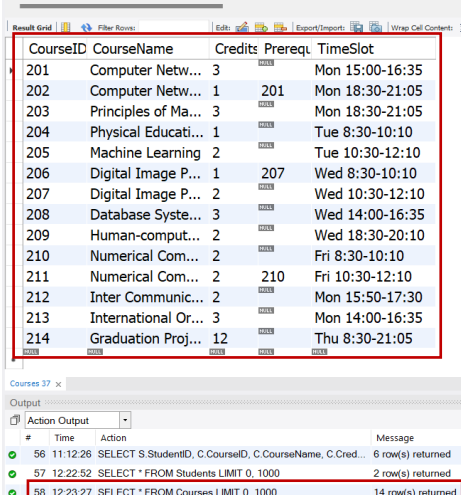
Students can access a complete list of available courses in the system. Course details include the course name, credit hours, prerequisites, and time slots.

#### MySQL Query:

```
SELECT * FROM Courses;
```

**Result:** Displays all course records, including CourseID, CourseName, Credits, and TimeSlot.

06 • **SELECT \* FROM Courses;**  
07



CourseID	CourseName	Credits	Prereq	TimeSlot
201	Computer Netw...	3		Mon 15:00-16:35
202	Computer Netw...	1	201	Mon 18:30-21:05
203	Principles of Ma...	3		Mon 18:30-21:05
204	Physical Educati...	1		Tue 8:30-10:10
205	Machine Learning	2		Tue 10:30-12:10
206	Digital Image P...	1	207	Wed 8:30-10:10
207	Digital Image P...	2		Wed 10:30-12:10
208	Database Syste...	3		Wed 14:00-16:35
209	Human-comput...	2		Wed 18:30-20:10
210	Numerical Com...	2		Fri 8:30-10:10
211	Numerical Com...	2	210	Fri 10:30-12:10
212	Inter Communic...	2		Mon 15:50-17:30
213	International Or...	3		Mon 14:00-16:35
214	Graduation Proj...	12		Thu 8:30-21:05

Courses 37 x

Output

#	Time	Action	Message
56	11:12:26	SELECT S.StudentID, C.CourseID, C.CourseName, C.Cred...	6 row(s) returned
57	12:22:52	SELECT * FROM Students LIMIT 0, 1000	2 row(s) returned
58	12:23:27	SELECT * FROM Courses LIMIT 0, 1000	14 row(s) returned

## 4.2 Registering for Courses

To enroll in a course, the system validates constraints such as prerequisite completion, credit limits, and time conflicts. Students can proceed with registration only if all conditions are satisfied.

**Python Logic:** The `check_constraints()` function ensures:

- Prerequisites for the selected course are completed.
- Total credits after registration do not exceed 21 credits.
- Time slots do not conflict with previously selected courses.

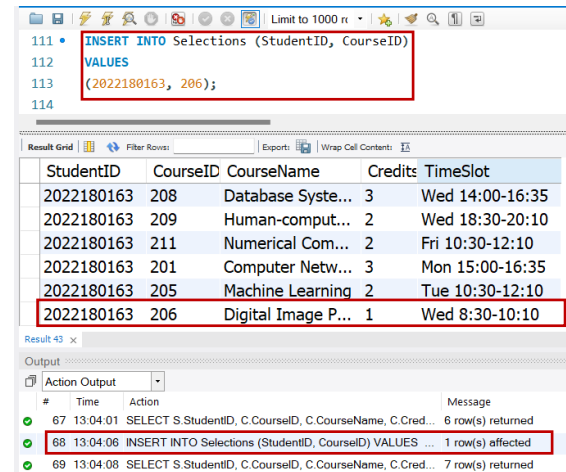


Figure: Adding course 206

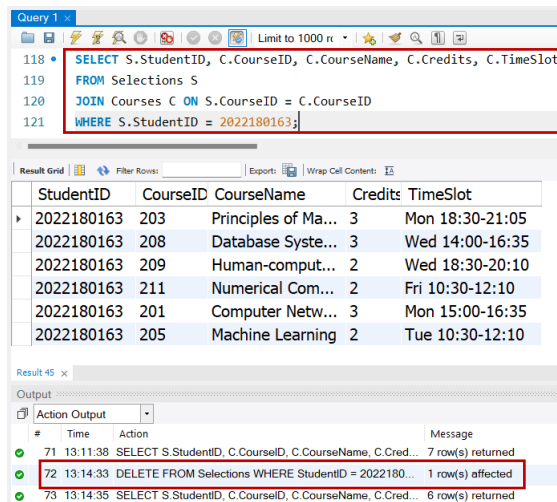


Figure: course 206 successfully dropped

## 4.3 Dropping a Course

Students can drop any registered course by removing its corresponding entry in the Selections table.

**MySQL Query:**

DELETE FROM Selections

WHERE StudentID = 2022180163 AND  
CourseID = 206;

**Result:** Confirms removal of the course from the student's registration list.

## 4.4 Viewing Registered Courses

Students can view their selected courses, along with course details, by joining the Selections table with the Courses table.

**MySQL Query:**

```
SELECT s.StudentID, c.CourseName,
c.Credits, c.TimeSlot
FROM Selections s
JOIN Courses c ON s.CourseID = c.CourseID
WHERE s.StudentID = 2022180163;
```

**Result:** Displays a list of courses registered by the student.

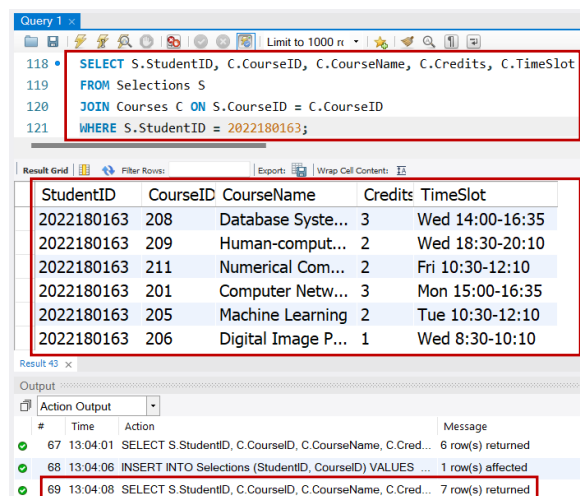


Figure: Chosen courses displayed in mySQL workbench

## 5. System Usage Guidelines for the Python code

### 1. System Overview

The Python application is a command-line interface that allows students to:

1. View available courses.
2. Register for courses after checking constraints such as prerequisites, credit limits, and time conflicts.
3. Remove registered courses.
4. View already selected courses.

The application interacts with the MySQL database to retrieve and manage data, ensuring all constraints are enforced.

### 2. Prerequisites

Before using the Python code, ensure the following:

1. Install Python 3 on the machine.
2. Install the mysql-connector-python library for database connectivity:

pip install mysql-connector-python

3. Ensure MySQL server is running, and the database `course_system` is properly set up with the required tables:
  - Students
  - Courses
  - Selections
4. Update the database connection details (host, user, password, and database) in the `connect_db` function of the Python code:

```
conn = mysql.connector.connect(  
    host="localhost",  
    user="root",  
    password="rootpassword",  
    database="course_system"  
)
```

### 3. Features and Usage

#### 3.1 Viewing Available Courses

The application fetches and displays all courses stored in the Courses table. Each course is shown with its ID, name, credits, and time slot.

**Steps:**

1. Run the Python script.
2. Select option 1 from the main menu to view courses.

**Code Logic:** The `view_courses()` function retrieves all rows from the Courses table and displays them in a formatted output.

### 3.2 Registering for a Course

Students can register for a course by entering their Student ID and the Course ID. The system validates the following constraints before adding the course:

1. **Prerequisites:** Ensures the prerequisite course has been completed.
2. **Credit Limit:** Checks that the student's total credits do not exceed 21.
3. **Time Conflicts:** Ensures no overlapping time slots exist.

#### Steps:

1. Select option 2 from the main menu.
2. Enter the Student ID and the Course ID you wish to register for.
3. The system will check constraints and, if valid, add the course to the Selections table.

#### Code Logic:

- The `add_course()` function calls `check_constraints()` to validate prerequisites, credit limits, and time conflicts.
- If all checks pass, the course is added to the Selections table using an INSERT query.

### 3.3 Removing a Course

Students can remove a previously registered course by entering their Student ID and the Course ID to be dropped.

#### Steps:

1. Select option 3 from the main menu.
2. Enter the Student ID and the Course ID of the course to remove.

**Code Logic:** The `remove_course()` function deletes the specified record from the Selections table using a DELETE query.

### 3.4 Viewing Selected Courses

Students can view the list of courses they have already registered for. The system retrieves course details from the Selections table and joins it with the Courses table.

#### Steps:

1. Select option 4 from the main menu.
2. Enter the Student ID to view all registered courses.

**Code Logic:** The `view_selected_courses()` function executes a SELECT query to join the Selections table with the Courses table and displays the results.

## 4. Running the Application

1. Save the Python code in a file named `app.py`.
2. Open a terminal or command prompt and navigate to the directory containing the script.
3. Run the script:

`python app.py`

4. Follow the menu options displayed in the terminal:
  - Option 1: View all available courses.
  - Option 2: Register for a course.
  - Option 3: Remove a registered course.
  - Option 4: View selected courses.
  - Option 5: Exit the application.



## 5. Example Usage

Below is an example session with the application:

### Menu Options:

1. View Courses
2. Add Course
3. Remove Course
4. View Selected Courses
5. Exit

### Example:

1. Selecting option 1 displays all courses:

ID: 201, Name: Computer Networks, Credits: 3, Time: Mon 14:00-15:40

ID: 202, Name: Computer Networks Lab, Credits: 1, Time: Mon 18:30-20:10

...

```
PS C:\Users\Duanm\Desktop\2nd_year\DSLab\lab\.vscode\.vscode\CourseSelectDatabase> python -u "c:\Users\Duanm\Desktop\2nd_year\DSLab\lab\.vscode\.vscode\CourseSelectDatabase\main.py"
```

```
1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
```

```
Enter your choice: 1
Database connected successfully!
Fetching courses...
Data fetched successfully!
```

```
Available Courses:
ID: 201, Name: Computer Networks, Credits: 3, Time: Mon 15:00-16:35
ID: 202, Name: Computer Networks Lab, Credits: 1, Time: Mon 18:30-21:05
ID: 203, Name: Principles of Marxism, Credits: 3, Time: Mon 18:30-21:05
ID: 204, Name: Physical Education III, Credits: 1, Time: Tue 8:30-10:10
ID: 205, Name: Machine Learning, Credits: 2, Time: Tue 10:30-12:10
ID: 206, Name: Digital Image Processing Lab, Credits: 1, Time: Wed 8:30-10:10
ID: 207, Name: Digital Image Processing, Credits: 2, Time: Wed 10:30-12:10
ID: 208, Name: Database Systems, Credits: 3, Time: Wed 14:00-16:35
ID: 209, Name: Human-computer Interaction, Credits: 2, Time: Wed 18:30-20:10
ID: 210, Name: Numerical Computation, Credits: 2, Time: Fri 8:30-10:10
ID: 211, Name: Numerical Computation Experiments, Credits: 2, Time: Fri 10:30-12:10
14:00-16:35
ID: 214, Name: Graduation Project, Credits: 12, Time: Thu 8:30-21:05
Database connection closed.
```

Figure: Viewing all courses in using Python in Visual Studio Code

2. Selecting option 2 (Add Course):

Enter your Student ID: 2022180163

Enter the Course ID to add: 214

```

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: 2
Enter your Student ID: 2022180163
Enter the Course ID to add: 214
Database connected successfully!
Database connected successfully!
Error: Credit limit of 21 exceeded

```

```

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: █

```

Figure: Based on the previous snapshots there are 14 credits worth of courses selected. If we try to select more than 21 courses, it will give us an error, in this case trying to add Graduation Project course 214 (12 credits).

3. Selecting option 5  
(View Selected  
Courses):  
Selected Courses for  
Student ID 2022180163:

```

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: 5
Enter your Student ID: 2022180163
Database connected successfully!

```

```

Courses Selected by Student ID 2022180163:
ID: 203, Name: Principles of Marxism, Credits: 3, Time: Mon 18:30-21:05
ID: 208, Name: Database Systems, Credits: 3, Time: Wed 14:00-16:35
ID: 209, Name: Human-computer Interaction, Credits: 2, Time: Wed 18:30-20:10
ID: 211, Name: Numerical Computation Experiments, Credits: 2, Time: Fri 10:30-12:10
ID: 201, Name: Computer Networks, Credits: 3, Time: Mon 15:00-16:35
ID: 205, Name: Machine Learning, Credits: 2, Time: Tue 10:30-12:10

```

Figure: Selected courses displayed in Visual Studio Code

4. Selecting  
option 3  
(Remove  
Course):  
Enter your  
Student  
ID:  
2022180163  
Enter the Course  
ID to remove: 205  
Course removed  
successfully!

```

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: 3
Enter your Student ID: 2022180163
Enter the Course ID to remove: 205
Database connected successfully!
Course removed successfully!

```

StudentID	CourseID	CourseName	Credits	TimeSlot
2022180163	203	Principles of Ma...	3	Mon 18:30-21:05
2022180163	208	Database Syste...	3	Wed 14:00-16:35
2022180163	209	Human-comput...	2	Wed 18:30-20:10
2022180163	210	Numerical Com...	2	Fri 8:30-10:10
2022180163	211	Numerical Com...	2	Fri 10:30-12:10
2022180163	201	Computer Netw...	3	Mon 15:00-16:35
2022180163	204	Physical Educati...	1	Tue 8:30-10:10

Figure: Machine Learning course (205) successfully

```

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: █

```

## 6. Error Handling

The system handles the following errors:

1. **Database Connection Errors:** If the connection fails, the application outputs an error message.
2. **Invalid Input:** Prompts the user to retry in case of invalid inputs (e.g., non-existent Course ID or Student ID).
3. **Constraint Violations:** Displays appropriate error messages if prerequisites are not satisfied, the credit limit is exceeded, or time conflicts exist.

## 6. Testing and Validation

The system was thoroughly tested using various scenarios to validate its functionality:

- Displaying all available courses to students.
- Enforcing prerequisite requirements during course registration.
- Preventing credit limit violations when selecting courses.
- Detecting time conflicts between selected courses.
- Allowing students to add, drop, and view their registered courses.

Each test case passed successfully, and the system performed as expected, maintaining robustness and reliability. The testcases will be proven by the following snapshots.

```
PS C:\Users\Duanm\Desktop\2nd_year\DSLab\lab\.vscode\.vscode\CourseSelectDatabase> python -u "c:\Users\Duanm\Desktop\2nd_year\DSLab\lab\.vscode\.vscode\CourseSelectDatabase\main.py"
```

1. View Courses

2. Add Course

3. Remove Course

4. Exit

5. View Selected Courses

Enter your choice: 1

Database connected successfully!

Fetching courses...

Data fetched successfully!

Available Courses:

ID: 201, Name: Computer Networks, Credits: 3, Time: Mon 15:00-16:35

ID: 202, Name: Computer Networks Lab, Credits: 1, Time: Mon 18:30-21:05

ID: 203, Name: Principles of Marxism, Credits: 3, Time: Mon 18:30-21:05

ID: 204, Name: Physical Education III, Credits: 1, Time: Tue 8:30-10:10

ID: 205, Name: Machine Learning, Credits: 2, Time: Tue 10:30-12:10

ID: 206, Name: Digital Image Processing Lab, Credits: 1, Time: Wed 8:30-10:10

ID: 207, Name: Digital Image Processing, Credits: 2, Time: Wed 10:30-12:10

ID: 208, Name: Database Systems, Credits: 3, Time: Wed 14:00-16:35

ID: 209, Name: Human-computer Interaction, Credits: 2, Time: Wed 18:30-20:10

ID: 210, Name: Numerical Computation, Credits: 2, Time: Fri 8:30-10:10

ID: 211, Name: Numerical Computation Experiments, Credits: 2, Time: Fri 10:30-12:10  
14:00-16:35

ID: 214, Name: Graduation Project, Credits: 12, Time: Thu 8:30-21:05

Database connection closed.

Figure: Viewing all courses in using Python in Visual Studio Code

CourseID	CourseName	Credits	Prerequisite	TimeSlot
201	Computer Networks	3	NULL	Mon 15:00-16:35
202	Computer Networks Lab	1	201	Mon 18:30-21:05
203	Principles of Marxism	3	NULL	Mon 18:30-21:05
204	Physical Education III	1	NULL	Tue 8:30-10:10
205	Machine Learning	2	NULL	Tue 10:30-12:10
206	Digital Image Processi...	1	207	Wed 8:30-10:10
207	Digital Image Processi...	2	NULL	Wed 10:30-12:10
208	Database Systems	3	NULL	Wed 14:00-16:35
209	Human-computer Inte...	2	NULL	Wed 18:30-20:10
210	Numerical Computation	2	NULL	Fri 8:30-10:10
211	Numerical Computatio...	2	210	Fri 10:30-12:10
212	Inter Communication	2	NULL	Mon 15:50-17:30
213	International Organiza...	3	NULL	Mon 14:00-16:35
214	Graduation Project	12	NULL	Thu 8:30-21:05

Figure: Viewing all courses in using in mySQL workbench

```

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: 2
Enter your Student ID: 2022180163
Enter the Course ID to add: 202
Database connected successfully!
Database connected successfully!
Error: Time conflict detected!

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: █

```

StudentID	CourseID	CourseName	Credits	TimeSlot
2022180163	203	Principles of Marxism	3	Mon 18:30-21:05
2022180163	205	Machine Learning	2	Tue 10:30-12:10
2022180163	208	Database Systems	3	Wed 14:00-16:35
2022180163	209	Human-computer Inte...	2	Wed 18:30-20:10
2022180163	210	Numerical Computation	2	Fri 8:30-10:10
2022180163	211	Numerical Computatio...	2	Fri 10:30-12:10
2022180163	201	Computer Networks	3	Mon 15:00-16:35

Figure: Trying to select Computer Networks Lab 202 (Mon 18:30-21:05), when Principles of Marxism of the same timeslot has already been chosen, in other words time conflict

```

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: 2
Enter your Student ID: 2022180163
Enter the Course ID to add: 214
Database connected successfully!
Database connected successfully!
Error: Credit limit of 21 exceeded

```

```

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: █

```

Figure: Based on the previous snapshots there are 14 credits worth of courses selected. If we try to select more than 21 courses, it will give us an error, in this case trying to add Graduation Project course 214 (12 credits).

Courses Selected by Student ID 2022180163:

```

ID: 203, Name: Principles of Marxism, Credits: 3, Time: Mon 18:30-21:05
ID: 205, Name: Machine Learning, Credits: 2, Time: Tue 10:30-12:10
ID: 208, Name: Database Systems, Credits: 3, Time: Wed 14:00-16:35
ID: 209, Name: Human-computer Interaction, Credits: 2, Time: Wed 18:30-20:10
ID: 210, Name: Numerical Computation, Credits: 2, Time: Fri 8:30-10:10
ID: 211, Name: Numerical Computation Experiments, Credits: 2, Time: Fri 10:30-12:10

```

```

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: 2
Enter your Student ID: 2022180163
Enter the Course ID to add: 206
Database connected successfully!
Prerequisite not satisfied. You need to complete Course ID: 207
Course cannot be added due to prerequisite requirements.

```

```

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: █

```

Figure: The Prerequisite requirement is not met. Suppose in order to chose the Lab course for Digital Image Processing (DIP), you should have chosen the theory course (207) first.

```

102 • SELECT S.StudentID, C.CourseID, C.CourseName, C.Credits, C.TimeSlot
103 FROM Selections S
104 JOIN Courses C ON S.CourseID = C.CourseID
105 WHERE S.StudentID = 2022180163;
106

```

Result Grid | Filter Rows: | Export: | Wrap Cell Contents: |

StudentID	CourseID	CourseName	Credits	TimeSlot
2022180163	203	Principles of Ma...	3	Mon 18:30-21:05
2022180163	205	Machine Learning	2	Tue 10:30-12:10
2022180163	208	Database Syste...	3	Wed 14:00-16:35
2022180163	209	Human-comput...	2	Wed 18:30-20:10
2022180163	210	Numerical Com...	2	Fri 8:30-10:10
2022180163	211	Numerical Com...	2	Fri 10:30-12:10
2022180163	201	Computer Netw...	3	Mon 15:00-16:35

Figure: Displaying already selected courses

1. View Courses	StudentID	CourseID	CourseName	Credits	TimeSlot
2. Add Course	2022180163	203	Principles of Ma...	3	Mon 18:30-21:05
3. Remove Course	2022180163	205	Machine Learning	2	Tue 10:30-12:10
4. Exit	2022180163	208	Database Syste...	3	Wed 14:00-16:35
5. View Selected Courses	2022180163	209	Human-comput...	2	Wed 18:30-20:10
Enter your choice: 2	2022180163	210	Numerical Com...	2	Fri 8:30-10:10
Enter your Student ID: 2022180163	2022180163	211	Numerical Com...	2	Fri 10:30-12:10
Enter the Course ID to add: 204	2022180163	201	Computer Netw...	3	Mon 15:00-16:35
Database connected successfully!	2022180163	204	Physical Educati...	1	Tue 8:30-10:10
Database connected successfully!					
Database connected successfully!					
Database connected successfully!					
Course added successfully!					

Figure: Course Physical Education ||| (204) successfully added

```

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: █

```

1. View Courses	StudentID	CourseID	CourseName	Credits	TimeSlot
2. Add Course	2022180163	203	Principles of Ma...	3	Mon 18:30-21:05
3. Remove Course	2022180163	208	Database Syste...	3	Wed 14:00-16:35
4. Exit	2022180163	209	Human-comput...	2	Wed 18:30-20:10
5. View Selected Courses	2022180163	210	Numerical Com...	2	Fri 8:30-10:10
Enter your choice: 3	2022180163	211	Numerical Com...	2	Fri 10:30-12:10
Enter your Student ID: 2022180163	2022180163	201	Computer Netw...	3	Mon 15:00-16:35
Enter the Course ID to remove: 205	2022180163	204	Physical Educati...	1	Tue 8:30-10:10
Database connected successfully!					
Course removed successfully!					

Figure: Machine Learning course (205) successfully removed.

```

1. View Courses
2. Add Course
3. Remove Course
4. Exit
5. View Selected Courses
Enter your choice: █

```

## **7. Conclusion**

The Course Registration System is an effective and user-friendly tool for managing student course enrollment. By automating the validation of prerequisites, credit limits, and time conflicts, it reduces manual effort and minimizes errors. The system's implementation using MySQL and Python ensures efficiency, scalability, and ease of use.

This guideline provides students and administrators with clear instructions for utilizing the system's features to register, drop, and view courses effortlessly.

## **8. Future Enhancements**

The system can be improved further by:

- Introducing a graphical user interface (GUI) for better user experience.
- Implementing role-based access control for students, instructors, and administrators.
- Adding automated notifications for course deadlines, conflicts and availability.
- Providing detailed course recommendations based on student preferences and history.

## 9. References

MySQL Documentation: <https://dev.mysql.com/>

Python MySQL Connector Library: <https://pypi.org/project/mysql-connector-python/>

Database Design Principles, *Database Management Systems* (Third Edition).